

Deep Generative Models: State Space Models, Selective State Space Models & MAMBA

Fall Semester 2025

René Vidal

Director of the Center for Innovation in Data Engineering and Science (IDEAS),
Rachleff University Professor, University of Pennsylvania
Amazon Scholar & Chief Scientist at NORCE



Taxonomy of Generative Models

What we've learned:
• MMs, HMMs, LDSs, RNNs

What we've learned:
• PPCA
• VAE

Deep Generative Models

Autoregressive models
(e.g., PixelCNN)

Flow-based models
(e.g., RealNVP)

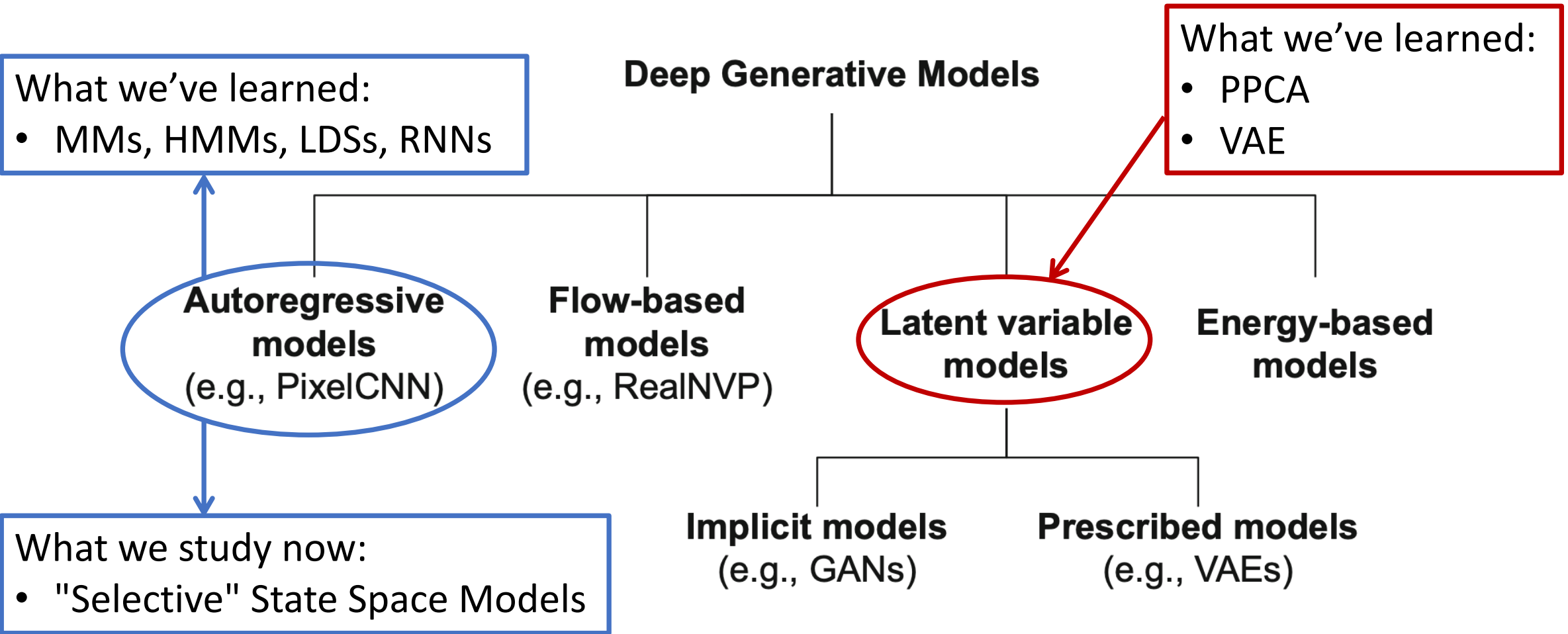
Latent variable models

Energy-based models

Implicit models
(e.g., GANs)

Prescribed models
(e.g., VAEs)

What we study now:
• "Selective" State Space Models



Selective State Space Models (Selective SSMs)

- State Space Models have overloaded meanings;
 - In particular, SSMs can mean linear dynamical systems

- LDSs (without noise)

$$\begin{aligned}z_t &= Az_{t-1} + Bx_t \\ y_t &= Cz_t\end{aligned}$$

- RNNs

$$\begin{aligned}z_t &= g(Az_{t-1} + Bx_t) \\ y_t &= f(Cz_t)\end{aligned}$$

- Selective SSMs

$$\begin{aligned}z_t &= A_t z_{t-1} + B_t x_t \\ y_t &= C_t z_t\end{aligned}$$

- A_t, B_t, C_t depend on input x_t
 - e.g., $A_t = s(x_t)$ for some learnable s
 - time-varying
- Similar, if not identical, to gating

Why Selective SSMs?

- LDSs
 - simple, theoretically grounded
 - not expressive enough
- RNNs
 - inefficient (can't be trained in parallel, can't even be efficiently unroll for 1K+ tokens)
 - gradient exploding & gradient vanishing
- Selective SSMs
 - efficient parallel algorithms
 - increased expressivity by incorporating gating or selection mechanism
 - avoid gradient exploding by careful parametrization

Problem Setup

$$z_t = A_t z_{t-1} + B_t x_t$$
$$y_t = C_t z_t$$

$$A_t = s_A(x_t), B_t = s_B(x_t), C_t = s_C(x_t)$$

s_A, s_B, s_C are learnable

- **Assumption:** A_t, B_t, C_t are diagonal
 - lead to scalar dynamics
 - crucial for efficiency
 - **Goal:** given initial state z_0 and $\{x_t, A_t, B_t, C_t\}_{t=1, \dots, T}$, compute y_1, \dots, y_T **efficiently**
 - this amounts to computing the forward pass of a SSM layer
 - during training, sequentially computing y_1, \dots, y_T won't be scalable for large T
- An example (yet practical) choice:
 - $s_A(x_t) = \text{diag}(\text{sigmoid}(\text{linear}(x_t)))$
 - $\text{sigmoid}(\xi) = \frac{1}{1 + \exp(-\xi)} \in (0, 1)$
 - prevents gradient explosion
 - crucial for stability
 - $\text{diag}(\cdot)$ makes A_t diagonal

Scalar Dynamics

- We now consider a case where everything is a scalar

$$z_t = a_t z_{t-1} + b_t x_t$$
$$y_t = c_t z_t$$

- **Goal:** given initial state z_0 and $\{x_t, a_t, b_t, c_t\}_{t=1, \dots, T}$, compute y_1, \dots, y_T **efficiently**
 - sequentially computing y_1, \dots, y_T won't be scalable for large T

- **Observation:**

- We only need to think about computing z_1, \dots, z_T with as much parallelism as possible
 - given all c_t 's and z_t 's, we can compute y_t 's in parallel
- We can compute $d_t := b_t x_t$ in parallel, so we

$$z_t = a_t z_{t-1} + d_t$$

Scalar Dynamics

$$z_t = a_t z_{t-1} + d_t$$

- Simplified Case 1: if $d_t = 0$, then we have

$$z_t = a_t a_{t-1} \cdots a_1 z_0$$

- we just need to compute $[a_1, a_1 a_2 \dots, a_1 a_2 \cdots a_T]$ (cumulative product)
- Simplified Case 2: if $a_t = 1$, then we have

$$z_t = z_0 + d_1 + \cdots + d_t$$

- we just need to compute $[d_1, d_1 + d_2 \dots, d_1 + d_2 + \cdots + d_T]$ (cumulative sum)

- In general, we can write the dynamics $z_t = a_t z_{t-1} + d_t$ as

$$\begin{bmatrix} z_t \\ 1 \end{bmatrix} = \begin{bmatrix} a_t & d_t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z_{t-1} \\ 1 \end{bmatrix}$$

- We just need to compute the cumulative product of 2×2 matrices $\begin{bmatrix} a_t & d_t \\ 0 & 1 \end{bmatrix}$

Associative Operator

$$\begin{bmatrix} z_t \\ 1 \end{bmatrix} = \begin{bmatrix} a_t & d_t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z_{t-1} \\ 1 \end{bmatrix}$$

- Matrix product, scalar product, scalar addition are all associative operators \oplus

$$a \oplus b \oplus c = (a \oplus b) \oplus c = a \oplus (b \oplus c)$$

- Since

$$\begin{bmatrix} a_t & d_t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_\tau & d_\tau \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} a_t a_\tau & a_t d_\tau + d_t \\ 0 & 1 \end{bmatrix}$$

we can define the associative operator \oplus this 2x2 matrix product as

$$(a_t, d_t) \oplus (a_\tau, d_\tau) = (a_t a_\tau, a_t d_\tau + d_t)$$

- **Why we need this notation:**

- For any associative operator \oplus , parallel algorithms exist to compute

$$[\xi_1, \xi_1 \oplus \xi_2 \dots, \xi_1 \oplus \xi_2 \oplus \dots \oplus \xi_T]$$

Hillis/Steele Associative Scan Algorithm

$$[\xi_1, \xi_1 \oplus \xi_2 \dots, \xi_1 \oplus \xi_2 \oplus \dots \oplus \xi_T]$$

step = 1, $p[i] = \xi_{i+1}$

while step < T: # assume T is some power of 2

for i in range(step, T): # can run in parallel with p processors

$$p[i] = p[i - step] \oplus p[i]$$

step *= 2

sequential work [while loop]: $\log(T)$
 total work: $\frac{T}{p} \cdot \log(T)$

Example (T=8): (the symbol \oplus is omitted in interest of space)

Iter	Step	$p[0]$	$p[1]$	$p[2]$	$p[3]$	$p[4]$	$p[5]$	$p[6]$	$p[7]$
0	1	ξ_1	ξ_2	ξ_3	ξ_4	ξ_5	ξ_6	ξ_7	ξ_8
1	2	ξ_1	$\xi_1\xi_2$	$\xi_2\xi_3$	$\xi_3\xi_4$	$\xi_4\xi_5$	$\xi_5\xi_6$	$\xi_6\xi_7$	$\xi_7\xi_8$
2	4	ξ_1	$\xi_1\xi_2$	$\xi_1\xi_2\xi_3$	$\xi_1\xi_2\xi_3\xi_4$	$\xi_2\xi_3\xi_4\xi_5$	$\xi_3\xi_4\xi_5\xi_6$	$\xi_4\xi_5\xi_6\xi_7$	$\xi_5\xi_6\xi_7\xi_8$
3	8	ξ_1	$\xi_1\xi_2$	$\xi_1\xi_2\xi_3$	$\xi_1\xi_2\xi_3\xi_4$	$\xi_1\xi_2\xi_3\xi_4\xi_5$	$\xi_1\xi_2\xi_3$ $\xi_4\xi_5\xi_6$	$\xi_1\xi_2\xi_3$ $\xi_4\xi_5\xi_6\xi_7$	$\xi_1\xi_2\xi_3\xi_4$ $\xi_5\xi_6\xi_7\xi_8$

Hillis/Steele Associative Scan Algorithm

$$[\xi_1, \xi_1 \oplus \xi_2 \dots, \xi_1 \oplus \xi_2 \oplus \dots \oplus \xi_T]$$

step = 1, $p[i] = \xi_{i+1}$

while step < T: # assume T is some power of 2

for i in range(step, T): # can run in parallel

$$p[i] = p[i - \text{step}] \oplus p[i]$$

step *= 2

Forward Pass of Selective SSMs (Vector Case)

s_A, s_B, s_C are learnable

$$z_t = A_t z_{t-1} + B_t x_t$$
$$y_t = C_t z_t$$

$$A_t = s_A(x_t), B_t = s_B(x_t), C_t = s_C(x_t)$$

- An example (yet practical) choice:
 - $s_A(x_t) = \text{diag}(\text{sigmoid}(\text{linear}(x_t)))$
 - $\text{sigmoid}(\xi) = \frac{1}{1 + \exp(-\xi)} \in (0, 1)$
 - prevents gradient explosion
 - crucial for stability
 - $\text{diag}(\cdot)$ makes A_t diagonal

- Given initial state z_0 and $\{x_t\}_{t=1, \dots, T}$
 1. compute the diagonal matrices A_t, B_t, C_t (for all t in parallel):
 - $A_t = s_A(x_t), B_t = s_B(x_t), C_t = s_C(x_t)$
 2. compute all vector outputs y_1, \dots, y_T via associative scan algorithms

Practical Use of Selective SSMs

[\[PDF\] Mamba: Linear-time sequence modeling with selective state spaces](#)



[A Gu, T Dao](#)

arXiv preprint arXiv:2312.00752, 2023 · [3dvar.com](#)

Abstract

Foundation models, now powering most of the exciting applications in deep learning, are almost universally based on the Transformer architecture and its core attention module. Many subquadratic-time architectures such as linear attention, gated convolution and recurrent models, and structured state space models (SSMs) have been developed to address Transformers' computational inefficiency on long sequences, but they have not performed as well as attention on important modalities such as language. We identify that

SHOW MORE ▾

☆ Save  Cite Cited by 5712 Related articles All 12 versions 

- Applicable to all RNN applications